Laboratorio 6: "Teclado Matricial y Comunicación Serial Asincrónica"

Objetivos.

- Experimentar con la conexión de dispositivos externos al microcontrolador.
- Aprender a adquirir datos desde un teclado matricial.
- Aprender a manejar la comunicación serial asincrónica.
- Manejar puertos del microcontrolador en forma bidireccional.

Preparación previa.

6.1 Teclado Matricial

Una de las formas más simples de ingresar información a un micro es a través de botones. Para ahorrar pines, se conectan los botones en forma matricial. A éstos se les llama Teclados Matriciales, los cuales pueden encontrarse en diversos equipos como Teléfonos, Alarmas y Microondas. Un teclado matricial tiene típicamente 12 ó 16 botones. En esta oportunidad utilizaremos un teclado de 16 botones, conectados en una matriz de 4 x 4. Cada uno de los botones tiene un terminal conectado a una fila y el otro conectado a una columna. Para saber qué tecla está presionada, se efectúa un barrido por filas para ver si ha habido un cambio en el estado de las columnas. En la figura 6.1 se muestra la conexión del teclado matricial a un puerto de la MSP430.

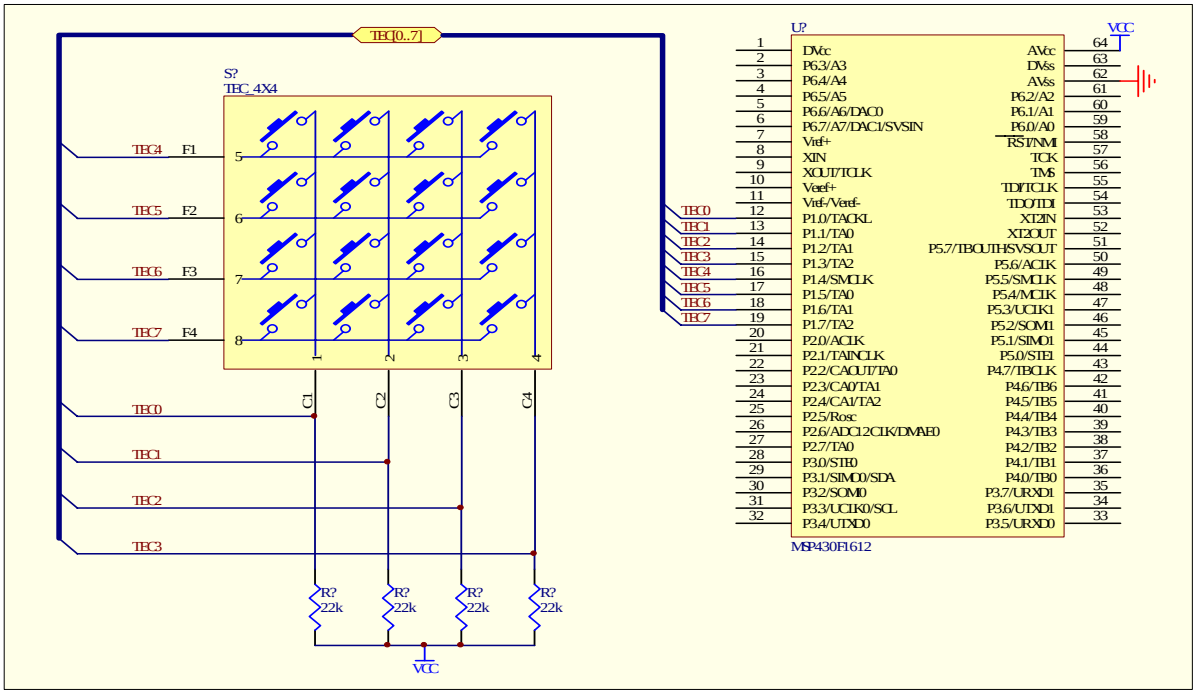

Figura 6.1: Señales de Interconexión Teclado Matricial

Existen 2 técnicas principales para hacer la lectura de cualquier tipo de botones: por interrupción y por encuesta. La primera consiste en que, al presionar el botón, el cambio de estado en el pin genera una interrupción. El problema con esta técnica consiste en que los botones tienen rebotes, lo que el microcontrolador puede interpretar como si se hubiese presionado el botón varias veces seguidas. Además, al soltar el botón el ruido puede generar nuevas interrupciones obteniendo por lo tanto una nueva lectura errónea. Por esta razón, se necesita eliminar el rebote ya sea por hardware ó por software.

La segunda técnica consiste en encuestar (pooling) el estado de los botones a intervalos fijos de entre 5 y 20 [ms] y comparar la lectura actual de los botones con la lectura anterior. Esta técnica tiene la ventaja de ser inmune al rebote de los botones, cuando el tiempo del rebote es menor al tiempo entre lecturas.

En la figura 6.1 los 4 bits menos significativos (lsb) del puerto 1 son entradas, mientras que los 4 bits más significativos (msb) son salidas. Normalmente el estado de los 4 pines de salida es alto y las resistencias de pull-up conectadas a los 4 bits menos significativos se encargan de mantener en un estado alto las columnas cuyos botones no se encuentran presionados.

Para hacer el barrido, se parte poniendo en 0 el primer bit (BIT4) correspondiente a la primera fila y se comprueba el estado de los 4 bits de entrada (BIT0:3). Si hay algún botón de la primera fila que esté presionado, se pondrá en 0 el estado de su respectiva columna. Si ningún botón está presionado, se vuelve a dejar la fila en 1 y se pone en 0 la fila siguiente y así sucesivamente hasta barrer todas las filas.

Luego de obtener la posición del botón presionado, debe compararse éste con el botón presionado en la lectura anterior. Si es el mismo, entonces no se ejecuta ninguna acción, puesto que el botón TODAVÍA se encuentra presionado. Puede también incluirse una variable que cuente el tiempo que se encuentra presionado el botón. Si por ejemplo, el botón lleva presionado más de 1 segundo, se asume como si el botón se volvió a presionar. Es el equivalente a dejar presionado cualquier botón en el teclado de un PC por más de 1 segundo.

Luego de reconocerse que se acaba de presionar un nuevo botón, debe traducirse esto al caracter o acción asociado al botón. Por ejemplo, al presionar la letra ‘A’ o el signo ‘#’, el programa deberá entrar a un menú y desplegarlo en el LCD, ó al presionar uno de los dígitos del ‘0’ al ‘9’, éste deberá ser impreso por pantalla.

Analice los programas `button_matrix.c` y `tic.c` que se entregan como ayuda para la captura de datos desde el teclado matricial (se encuentran en la página de la asignatura).

Programa 6.1

Cree un programa que vaya mostrando en la 2ª línea del display LCD un número introducido a través de teclado matricial, correspondiente a los dígitos. Si se presiona la tecla # (Enter), la MSP deberá copiar el texto de la fila 2 en la fila 1, borrar la fila 2 y poner el cursor al principio de la fila 2 (utilice ‘\n’ del lab4). Si se presiona la tecla C (Clear), deberá borrarse la fila 2 y poner el cursor al principio de ésta. Al presionar la tecla B (back space), deberá borrarse el último dígito ingresado. Para el resto de las teclas

proponga alguna acción a su elección. Utilice un timer para generar un barrido periódico del teclado matricial.

6.2 Comunicación Serial

En la MSP430 existe por lo menos un módulo USART. En el caso específico de los modelos MSP430F149 y MSP430F1612, se cuenta con dos módulos.

Este módulo se puede utilizar en dos tipos de comunicación serial: como UART (comunicación asincrónica) o como USART (comunicación sincrónica) en modo SPI ó I^2C. Toda la documentación necesaria para su utilización se encuentra en el manual del usuario. Es imprescindible su lectura.

En comunicación UART, un frame de datos comienza cuando ocurre un canto de bajada, luego se envía un bit de start, con nivel bajo. Luego los bits de datos, con el menos significativo (lsb) primero; y el más significativo (msb) al final. Luego termina el frame con un bit de stop en nivel alto. En la figura 6.2 se muestra un frame de transmisión serial

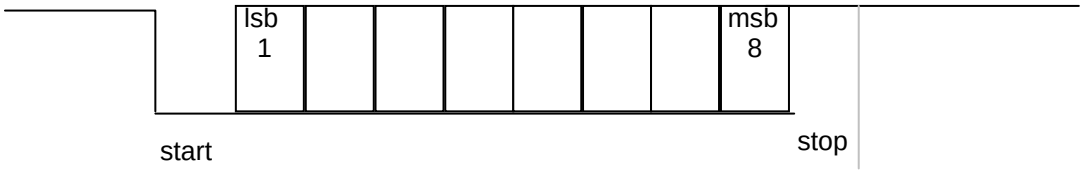

Figura 6.2: Frame de una comunicación Serial Asincrónica

Puede programarse un frame con uno, 1.5 ó 2 bits de stop. La paridad de la comunicación también puede ser programada. La velocidad de transmisión se mide en baudios, que está definido como el número de símbolos (en este caso 1 símbolo = 1 bit) por segundo. Las velocidades usuales son: 300, 1200, 2400, 4800, 9600, 19.200, 38.400 hasta 115.200 en algunas puertas seriales.

Estudie el programa 6.2 y responda las siguientes preguntas:

a) ¿Qué valor deben tener los registros U0BR1 y U0BR0? Cuál es el máximo error de bit configurando sólo estos 2 registros?

b) ¿Para qué sirve el registro U0MCTL? ¿Es necesario configurarlo para este caso?

c) ¿Qué hace el registro U0TXBUF? ¿Cómo se sabe si está vacío u ocupado?

d) ¿Cuál es la diferencia entre el flag UTXIFG0 (registro IFG1) y el flag TXEPT (registro U0TCTL)?

Programa 6.2

```
#include <msp430x16x.h>  // Depende del uC que Ud. esté ocupando.

char mensaje[10] = "Hola Mundo";
int i = 0,j = 0;

void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  P5SEL |= 0x70;      // Se dispone de los 3 relojes del sistema
  P5DIR |= 0x70;      // en los bits 4, 5 y 6 de la puerta 5
                      // (MCLK, SMCLK y ACLK)
  P3SEL |= 0x30;
  ME1 |= UTXE0 + URXE0;
  UCTL0 |= CHAR;
  //UTCTL0 |= SSEL1;
  UTCTL0 = 0x10;
  UBR00 = 0x03;
  UBR10 = 0x00;
  UMCTL0 = 0x4A;
  UCTL0 &= ~SWRST;

  while(1)
  {
    while (!(IFG1 & UTXIFG0));
    TXBUF0 = mensaje[i++];
    while (j < 1000) j++;
    j = 0;
    if (i >= 10) i = 0;
  }
}
```

Analice el programa 6.3 que permite capturar la recepción de datos vía una rutina de servicio de interrupción.

Programa 6.3

```
#include <msp430x16x.h>
#include <stdio.h>

char t = 0x0;

void main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  _EINT();
  P1DIR |= 0x01;
  P2DIR |= 0x01;
  P3SEL |= 0x30;
  ME1 |= UTXE0 + URXE0;
  UCTL0 |= CHAR;
  UTCTL0 |= SSEL0;
  UBR00 = 0x0D;
  UBR10 = 0x00;
  UMCTL0 = 0x6B;
```

```
  UCTL0 &= ~SWRST;
  IE1 |= URXIE0;

  while (!(IFG1 & UTXIFG0));
  TXBUF0 = t;

  for (;;)
  {
    t = getchar();
    while (!(IFG1 & UTXIFG0));
    TXBUF0 = t;
  }
}

#pragma vector=UART0RX_VECTOR
__interrupt void usart0_rx (void)
{
   /*Completar*/
}
```

En el Laboratorio.

6.1 Muestre el funcionamiento del programa 6.1.

6.2 Utilizando el programa 6.2:

- Cambie el programa de manera de poder transmitir a 4800 y a 9600 bps. Mida con osciloscopio.
- Cambie el programa de manera de utilizar dos bits de stop.
- Cambie el programa para transmitir caracteres de 7 bits sin paridad a 2400 bps. Mida con osciloscopio.

6.3 Considerando el programa 6.3, complete la rutina de servicio de interrupción serial de tal modo de retransmitir el carácter ingresado por consola, indicando en el display la tecla presionada y la cantidad de retransmisiones. Las retransmisiones ocurren solamente mientras un pin de control (a elección) se encuentre en alto, en caso contrario deben cesar. El display muestra siempre, el último estado de las retransmisiones. Considere para esto un funcionamiento con:

i. Un loopback vía software.
ii. Un loopback físico.

6.4 Diseñe un programa que captura la entrada del teclado matricial de acuerdo al programa de la sección 6.1, y la envía por el puerto serie. Configure para 4800 bps o 9600 bps dependiendo del estado de un pin. Use paridad par y 1 bit de stop. Habilite la rutina de recepción de datos seriales e interprete los datos como si fueran introducidos por el teclado matricial (programa 6.1). Para las pruebas conecte el pin RX con el TX. Conecte su circuito con el de otro grupo y pruebe el programa. Compruebe que ambos están utilizando la misma configuración para la UART.